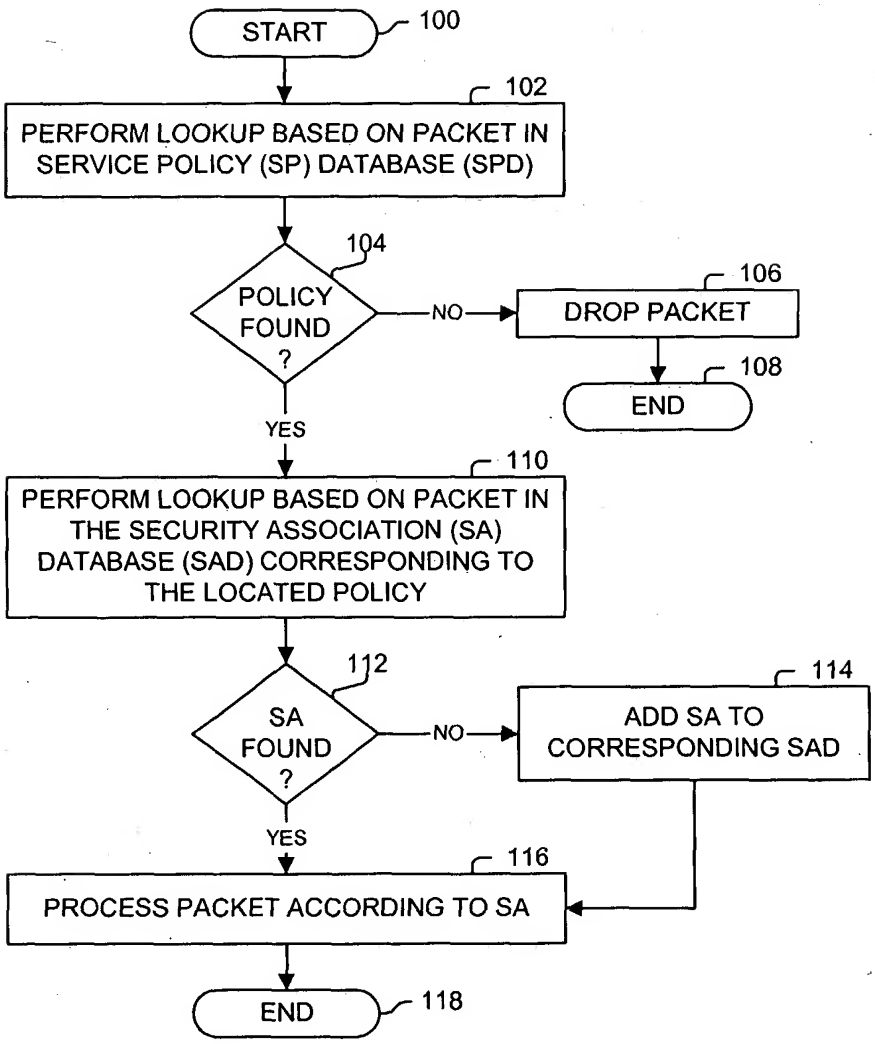


FIGURE 1
(PRIOR ART)

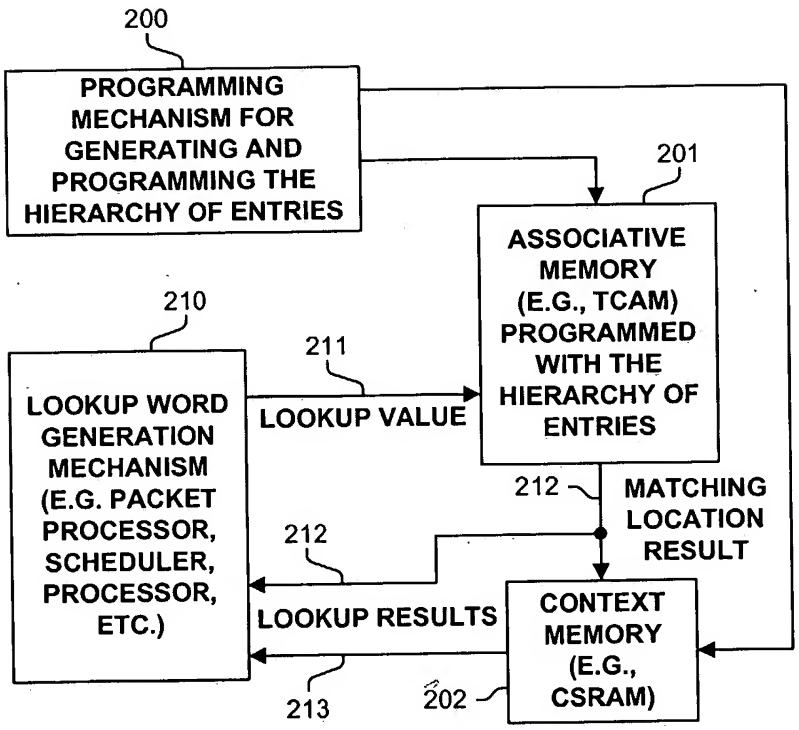

200
PROGRAMMING MECHANISM FOR GENERATING AND PROGRAMMING THE HIERARCHY OF ENTRIES
201
ASSOCIATIVE MEMORY (E.G., TCAM) PROGRAMMED WITH THE HIERARCHY OF ENTRIES
210
LOOKUP WORD GENERATION MECHANISM (E.G. PACKET PROCESSOR, SCHEDULER, PROCESSOR, ETC.)
211
LOOKUP VALUE
212
MATCHING LOCATION RESULT
212
LOOKUP RESULTS
213
202
CONTEXT MEMORY (E.G., CSRAM)

FIGURE 2A

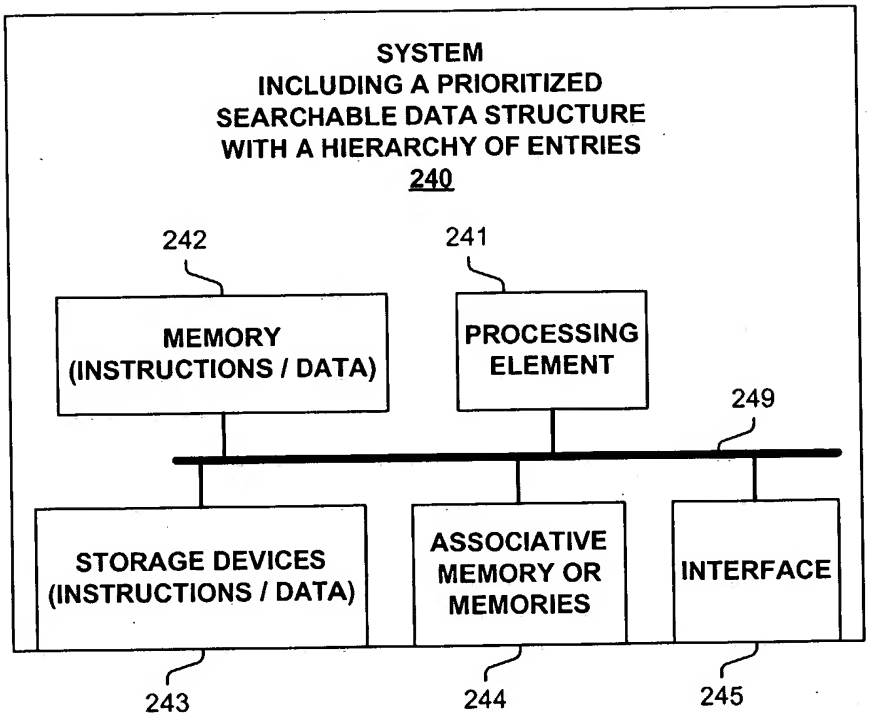

SYSTEM
INCLUDING A PRIORITIZED
SEARCHABLE DATA STRUCTURE
WITH A HIERARCHY OF ENTRIES
240
242
241
MEMORY
(INSTRUCTIONS / DATA)
PROCESSING
ELEMENT
249
STORAGE DEVICES
(INSTRUCTIONS / DATA)
ASSOCIATIVE
MEMORY OR
MEMORIES
INTERFACE
243
244
245

FIGURE 2B

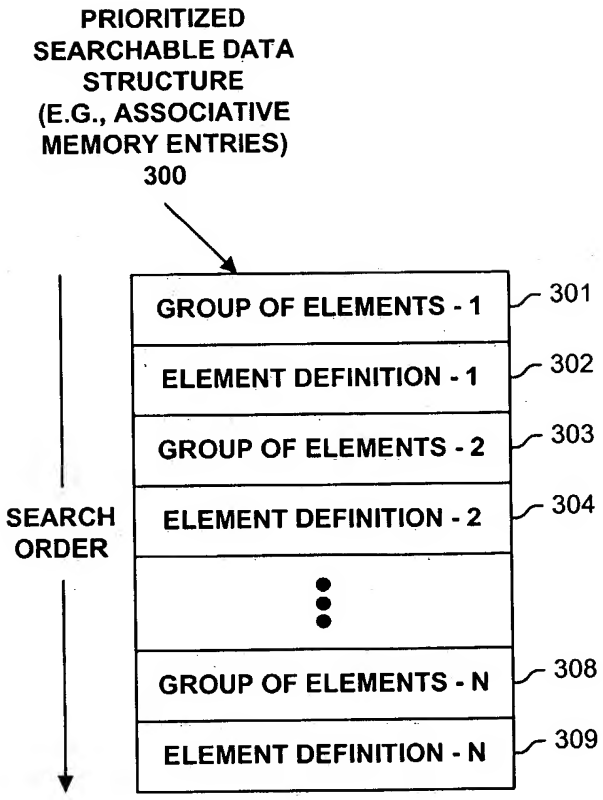

PRIORITIZED SEARCHABLE DATA STRUCTURE (E.G., ASSOCIATIVE MEMORY ENTRIES)
300
SEARCH ORDER
GROUP OF ELEMENTS - 1
301
ELEMENT DEFINITION - 1
302
GROUP OF ELEMENTS - 2
303
ELEMENT DEFINITION - 2
304
GROUP OF ELEMENTS - N
308
ELEMENT DEFINITION - N
309

FIGURE 3A

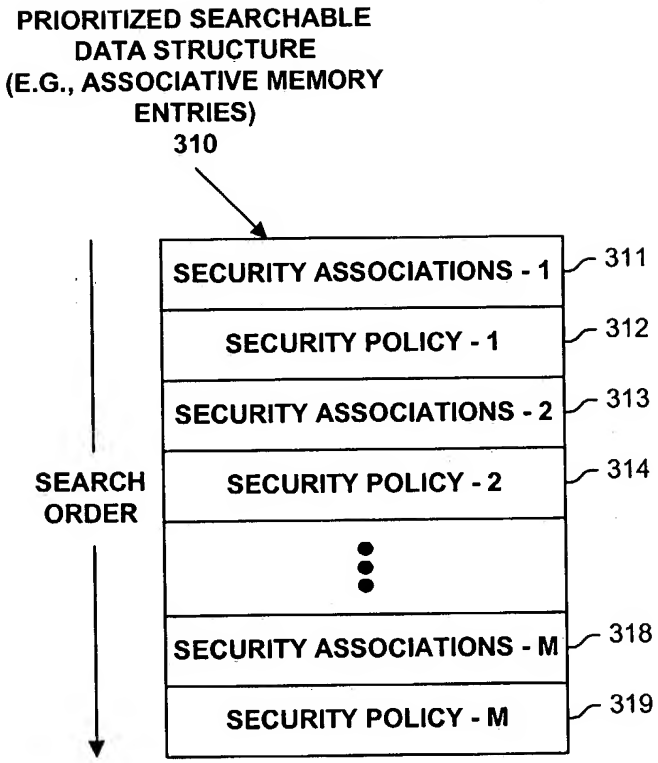

PRIORITIZED SEARCHABLE DATA STRUCTURE (E.G., ASSOCIATIVE MEMORY ENTRIES)
310
SECURITY ASSOCIATIONS - 1
311
SECURITY POLICY - 1
312
SECURITY ASSOCIATIONS - 2
313
SECURITY POLICY - 2
314
SEARCH ORDER
SECURITY ASSOCIATIONS - M
318
SECURITY POLICY - M
319

FIGURE 3B

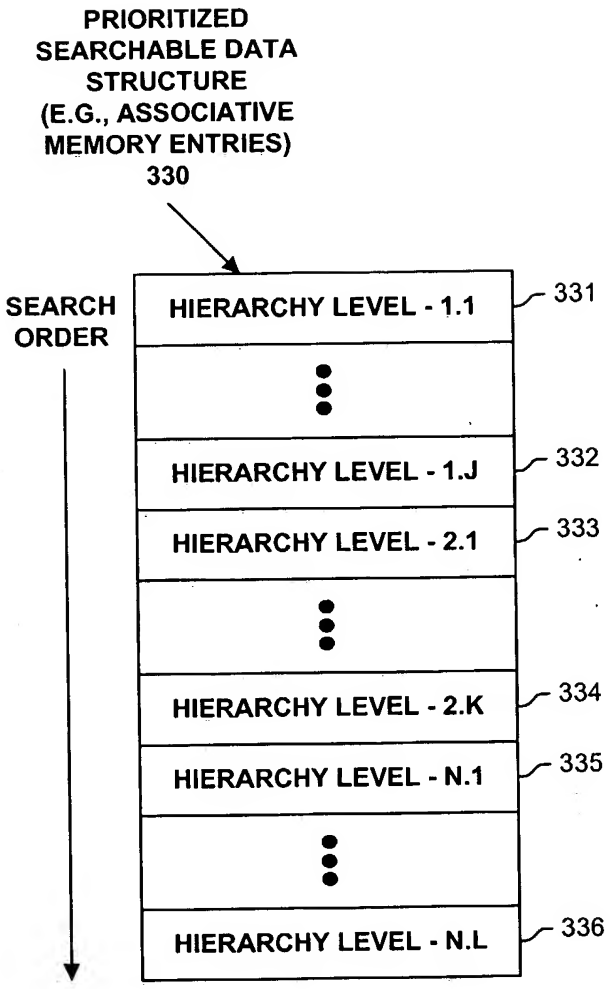
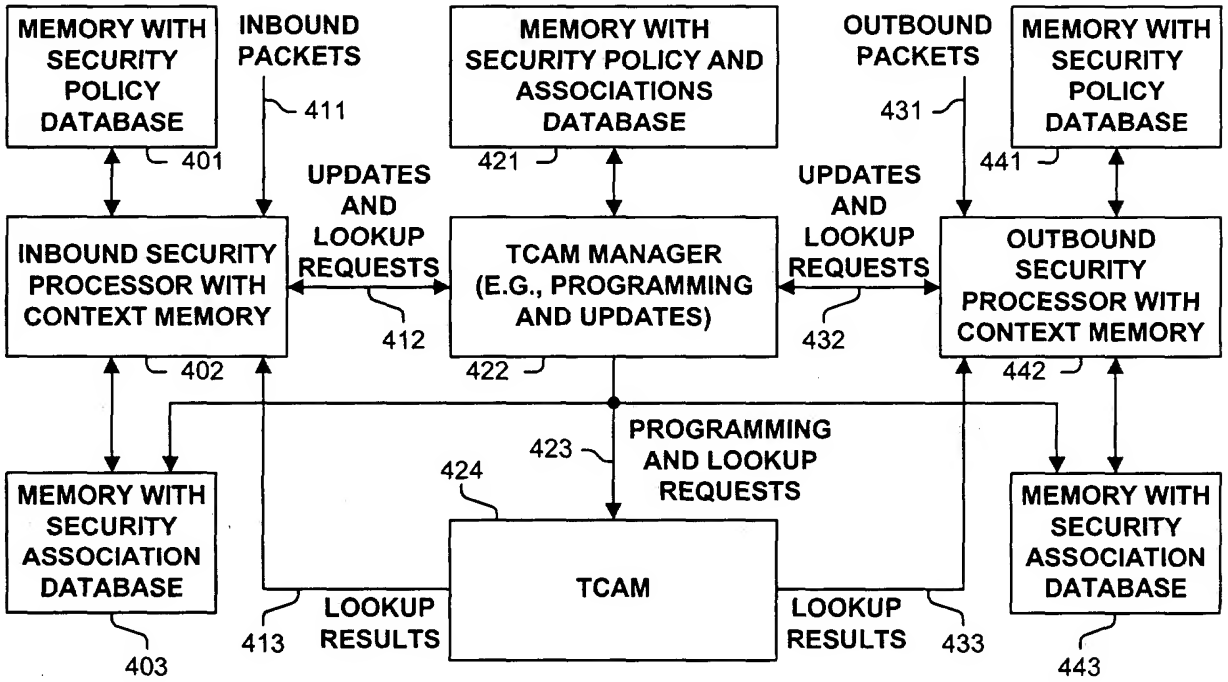

PRIORITIZED SEARCHABLE DATA STRUCTURE (E.G., ASSOCIATIVE MEMORY ENTRIES)
330
SEARCH ORDER
HIERARCHY LEVEL - 1.1
331
HIERARCHY LEVEL - 1.J
332
HIERARCHY LEVEL - 2.1
333
HIERARCHY LEVEL - 2.K
334
HIERARCHY LEVEL - N.1
335
HIERARCHY LEVEL - N.L
336

FIGURE 3C



IPSEC MECHANISM
FIGURE 4

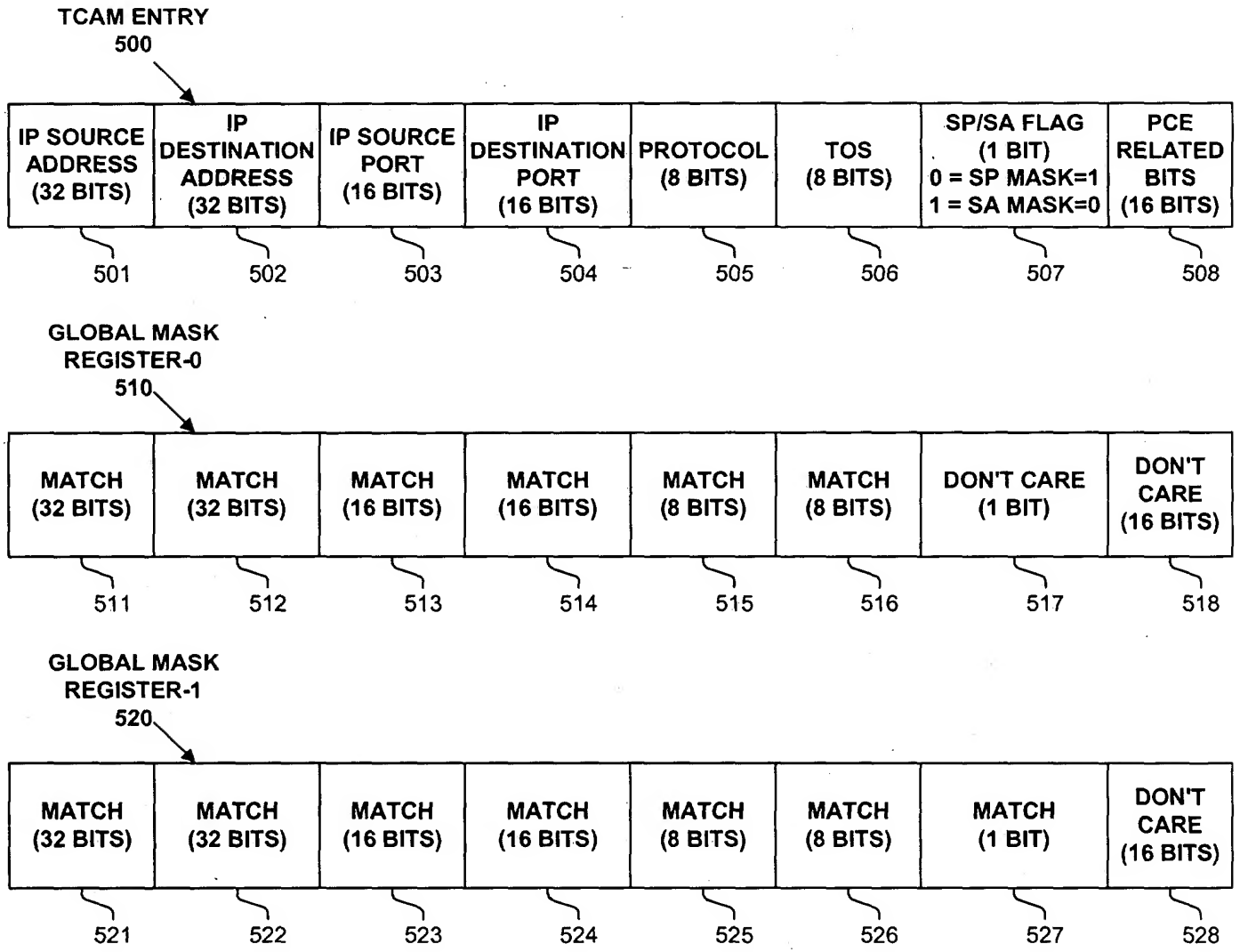
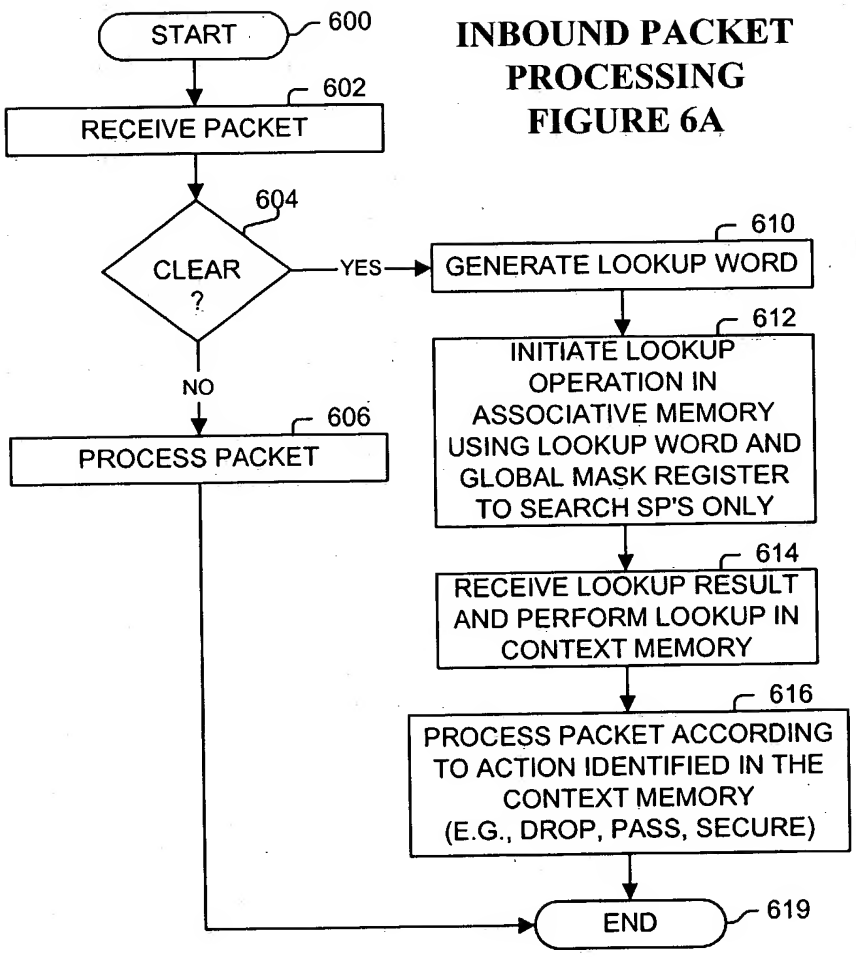

TCAM ENTRY
500
IP SOURCE ADDRESS (32 BITS)
IP DESTINATION ADDRESS (32 BITS)
IP SOURCE PORT (16 BITS)
IP DESTINATION PORT (16 BITS)
PROTOCOL (8 BITS)
TOS (8 BITS)
SP/SA FLAG (1 BIT) 0 = SP MASK=1 1 = SA MASK=0
PCE RELATED BITS (16 BITS)
501
502
503
504
505
506
507
508
GLOBAL MASK REGISTER-0
510
MATCH (32 BITS)
MATCH (32 BITS)
MATCH (16 BITS)
MATCH (16 BITS)
MATCH (8 BITS)
MATCH (8 BITS)
DON'T CARE (1 BIT)
DON'T CARE (16 BITS)
511
512
513
514
515
516
517
518
GLOBAL MASK REGISTER-1
520
MATCH (32 BITS)
MATCH (32 BITS)
MATCH (16 BITS)
MATCH (16 BITS)
MATCH (8 BITS)
MATCH (8 BITS)
MATCH (1 BIT)
DON'T CARE (16 BITS)
521
522
523
524
525
526
527
528

FIGURE 5A

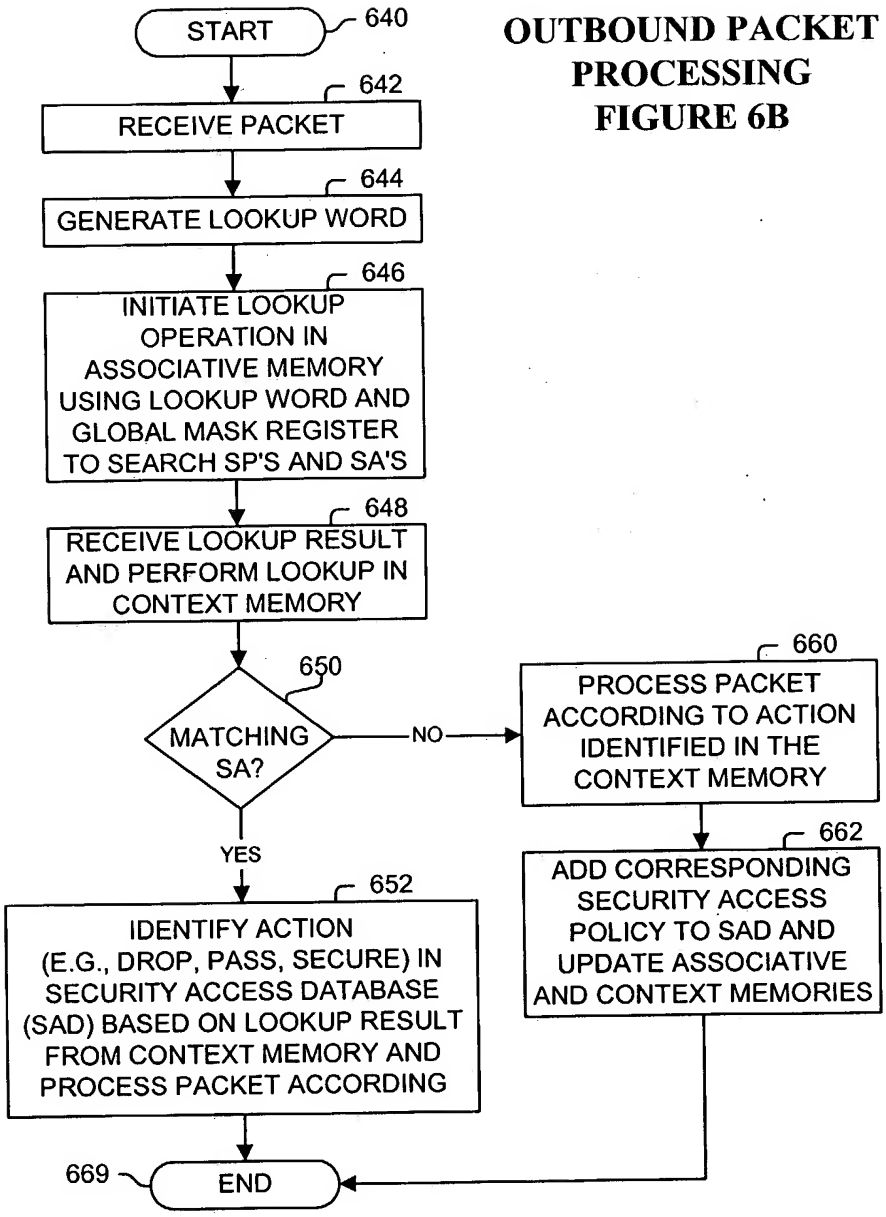
```
Arguments: highIncl, lowIncl; // range is [highIncl, lowIncl]

// Keep four stacks: two for results (value and masks), two for work in progress: bit and prefix
// for 16 bit fields, the results can be up to 30 TCAM entries, and the wip stacks must be 17 entries
// deep. (seems to be 2(N-1) worst case entries, but I have not generated a proof!)

push (prefixStack, 0), push (bitStack, precision); // 16 for port numbers
do {
        prefix = pop (prefixStack), bit = pop (bitStack);
        mask = (1 << bit) - 1;
        highTry = prefix | mask, lowTry = prefix & ~mask;
        if (highTry <= highIncl && lowTry >= lowIncl) { // entry covers a subset of range - save it
                push (valueStack, prefix), push (maskStack, mask);
        } else if (lowTry > highIncl || highTry < lowIncl) { // entry covers a disjoint set - forget it
                // do nothing
        } else { // entry covers some value inside and outside range - split it
                push (prefixStack, prefix), push (bitStack, bit - 1);
                push (prefixStack, prefix | 1 << (bit - 1)), push (bitStack, bit - 1);
        }
} while (StackNotEmpty (prefixStack));
```

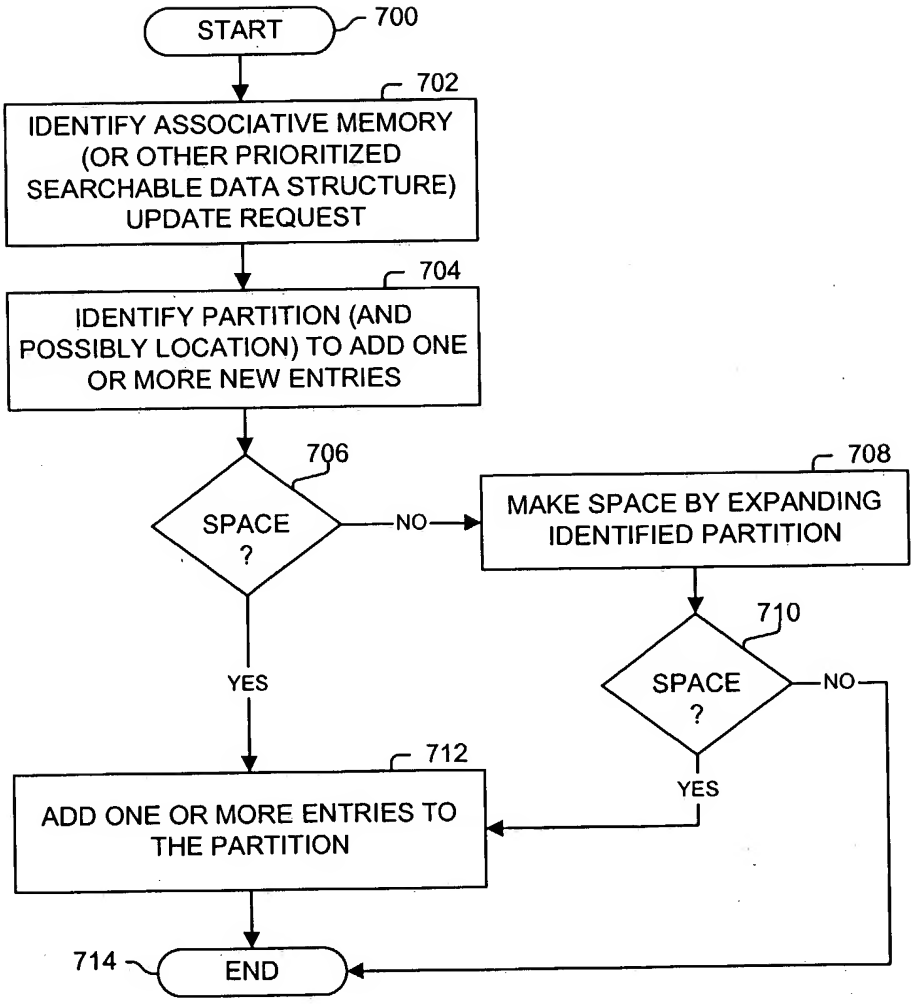
SPLITTING ENTRIES INTO MULTIPLE ENTRIES
FIGURE 5B



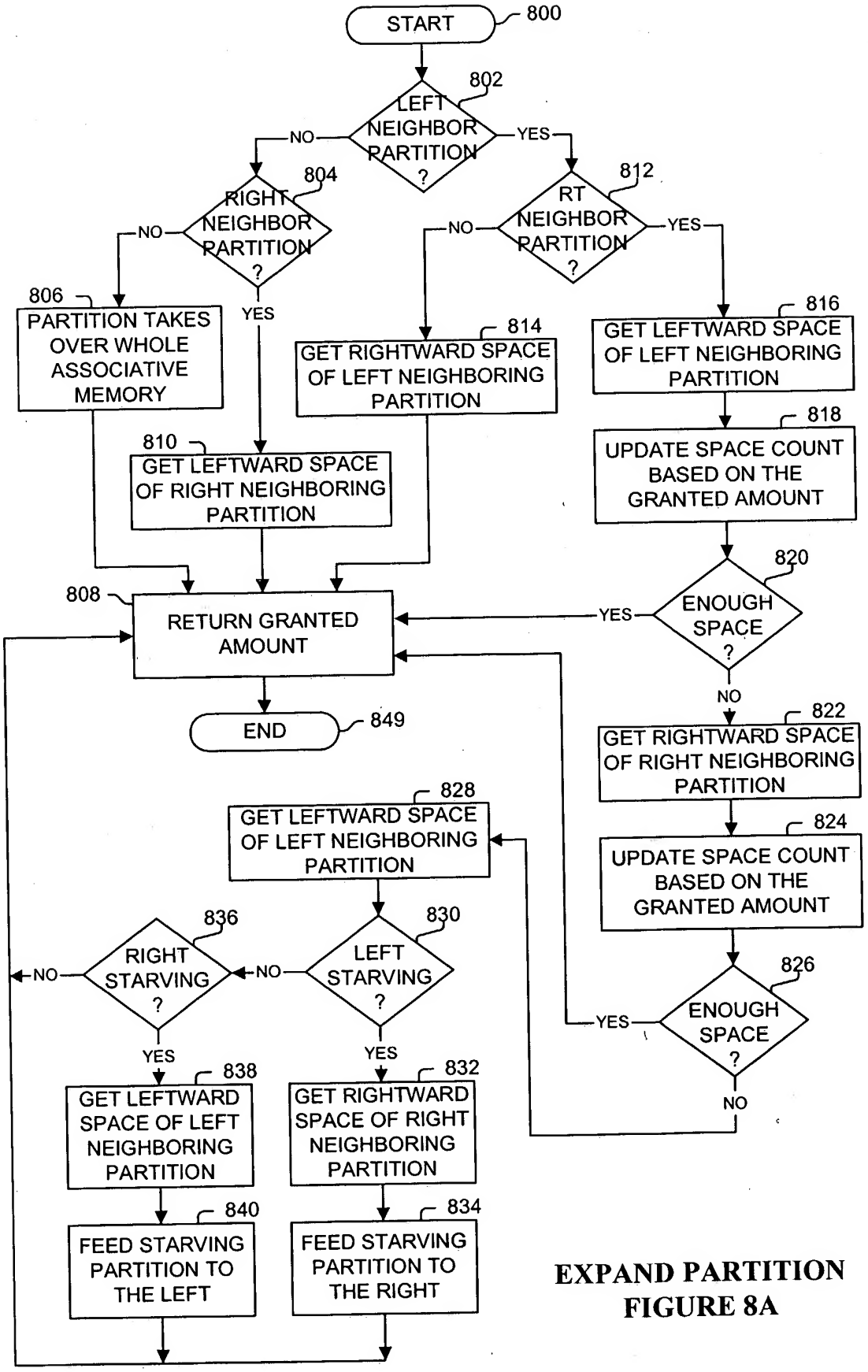
INBOUND PACKET PROCESSING
FIGURE 6A

OUTBOUND PACKET PROCESSING
FIGURE 6B

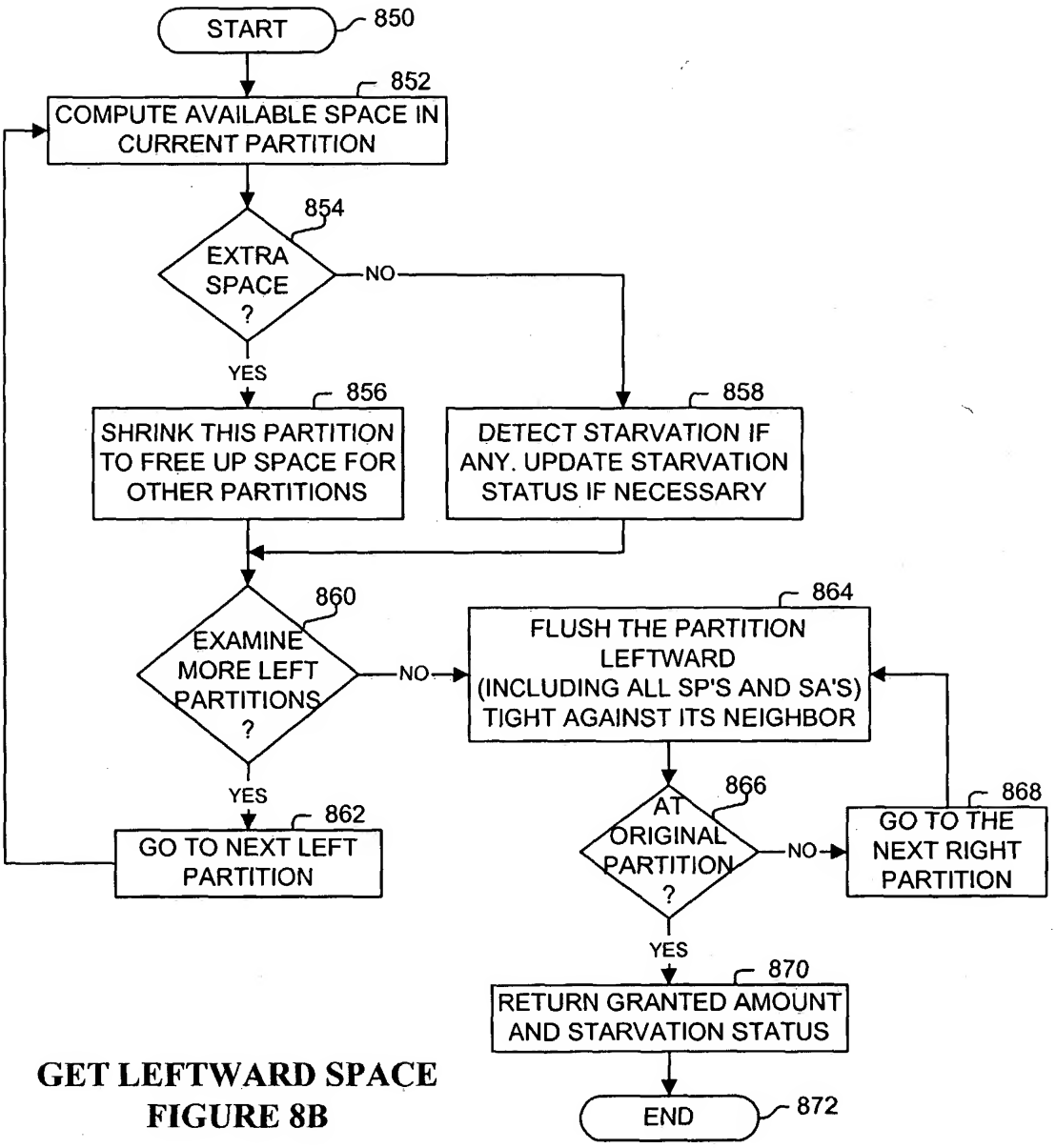




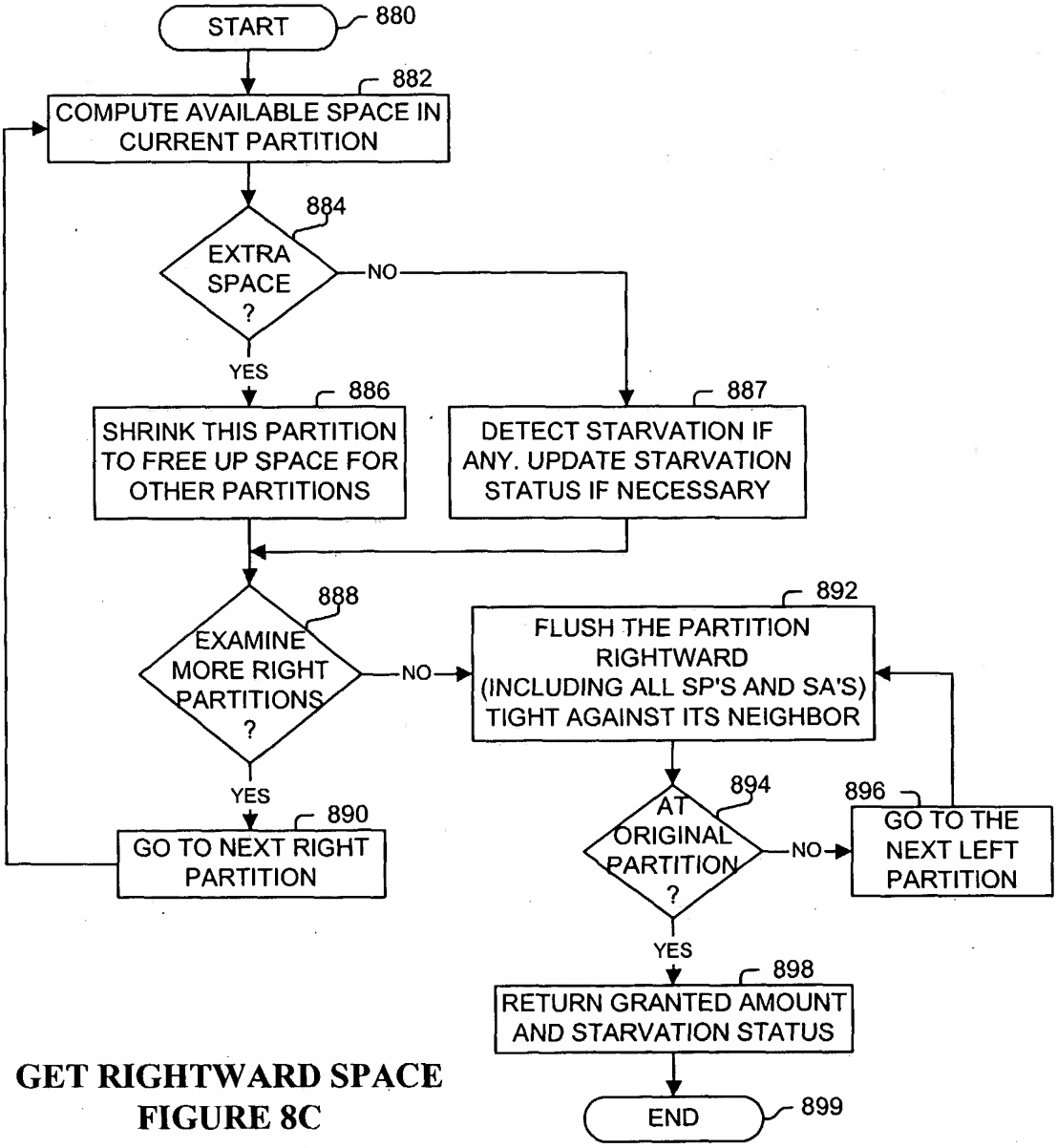
ADDING ENTRY
FIGURE 7



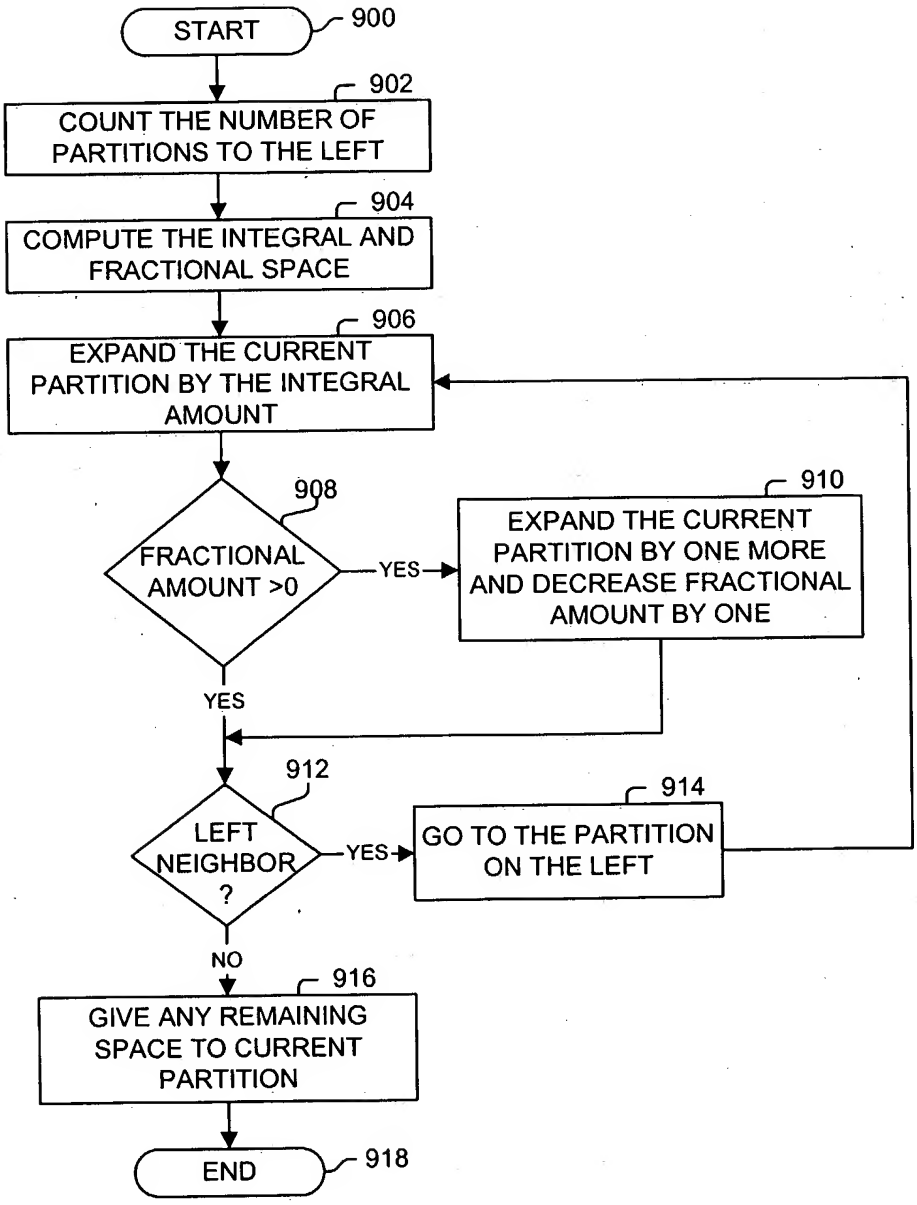
EXPAND PARTITION
FIGURE 8A



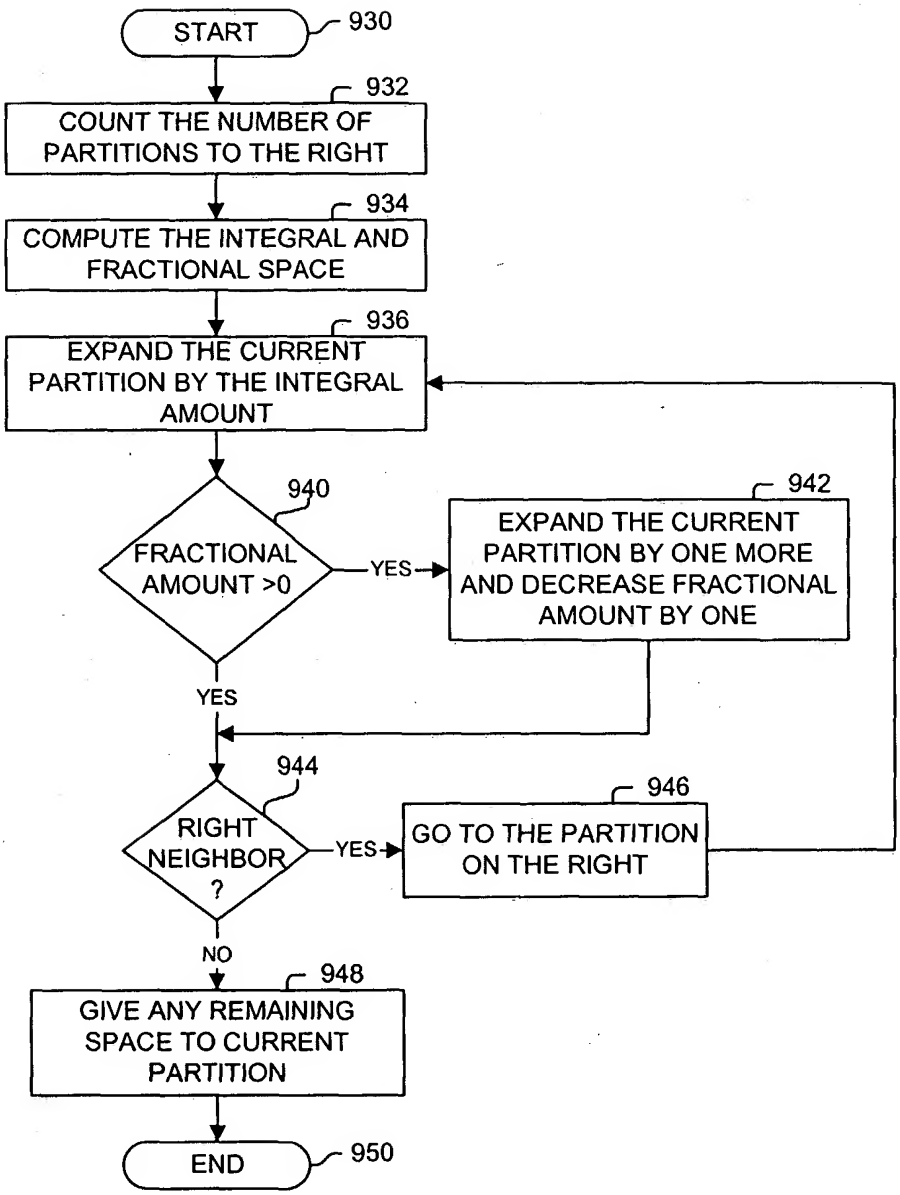
GET LEFTWARD SPACE
FIGURE 8B



GET RIGHTWARD SPACE
FIGURE 8C



FEED STARVING LEFT PARTITION
FIGURE 9A



FEED STARVING LEFT PARTITION
FIGURE 9B